Taming the Flying Cable Monster: A Topology Design and Optimization Framework for Data-Center Networks

Jayaram Mudigonda
Praveen Yalagandula
Jeff Mogul

HP Labs, Palo Alto, CA
Wiring Data Centers: A complex problem

Goal: design a cost-effective network for a large data center
This paper

Introduces a new research area: datacenter topology design and wiring
- Characterizes the problem and exposes several challenges
- Presents a novel framework, Perseus, for datacenter network design
- Describes the workflow for finding a cost-effective network
- Solves several novel optimization problems

Disclaimers: This paper does not
- Quantify precise costs of different network designs
  - Please do not believe the cost numbers we present in the paper
- Compare general merits of different topologies
- Consider all dimensions of the design space
Outline

Introduction
Problem
Perseus Framework
   Workflow, Topologies, Optimizations
Results
Further steps
Summary
Topologies

Trends:
- Datacenters are becoming larger and larger
- Need high bisection bandwidth: E.g., Map-Reduce, VM placement

Traditional topologies (tree-like) are not scalable
- Core switch is the bottleneck for bandwidth

Data-center networks need newer multi-path topologies
- That achieve high bisection bandwidth with limited port count switches
- E.g., FatTree, HyperX, Bcube

So far these topologies have not been feasible but for the advent of
- Cheap high speed high port count switches
- Multi-path forwarding techniques: VL2, SPAIN, PortLand, etc.
Problem: Design space too large for humans

Many topologies to choose from

- Several different topology families

  - Fat Tree
  - HyperCube
  - Clique
  - HyperX

- Several free parameters \(\rightarrow\) large number of choices within each family
  - Switch port count
  - Number of servers per edge-switch
  - Link speeds

Previous topology work: Mostly focused on a few logical metrics

- Bisection bandwidth, Maximum number of hops, etc.

But in practice, wiring becomes a complex problem
Wiring is a complex problem

Goal is to maximize performance at minimum cost

<table>
<thead>
<tr>
<th>Bisection Bandwidth</th>
<th>Capital</th>
<th>Operational</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worst-case Latency</td>
<td>Switches</td>
<td>Power</td>
</tr>
<tr>
<td>Reliability</td>
<td>Cables</td>
<td>SKUs</td>
</tr>
<tr>
<td>Serviceability</td>
<td>Racks</td>
<td>Administration</td>
</tr>
<tr>
<td>Expandability</td>
<td>Physical Space</td>
<td>(regular maintenance, fixing faults)</td>
</tr>
</tbody>
</table>
Real world constraints

- Face-plate size restricts number of switch connectors
- Cross-aisle cable trays cannot be over every rack
- Rack plenum restricts the size of cable bundle
- Cable length restrictions:
  - e.g., copper 10GbE has max range of ~10m
10GbE Cable Prices

<table>
<thead>
<tr>
<th>Length (Meters)</th>
<th>SFP+ Copper</th>
<th>QSFP+ Copper</th>
<th>QSFP+ Optical</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50</td>
<td>100</td>
<td>200</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>200</td>
<td>400</td>
</tr>
<tr>
<td>100</td>
<td>500</td>
<td>1000</td>
<td>2000</td>
</tr>
</tbody>
</table>

Sources:
www.cablesondemand.com
www.elpeus.com
Related work - I

Classical topology analysis

- Mainly focused on bisection bandwidth & hop counts
  - Ahn et al. 2009: find HyperX topology with min # of switches that achieve a given bisection BW
- Cabling complexity/cost was not considered

Placement and routing problems are similar to those in VLSI at a high level

- But different in details
Related work - II

Popa et al 2010: Compared the cost of different DC network architectures
- Did not focus on cost minimization in each topology family
- Did not consider placement optimization problem
- Assumed simpler model for cable costs

Farrington et al 2009: Analyzed cabling issues for FatTree networks
- Upper level switches and levels consolidated
- Design using merchant silicon, with cables as traces on circuit boards
Outline

Introduction

Problem

Perseus Framework

  Workflow, Topologies, Optimizations

Results

Further steps

Summary
Perseus

Framework to assist network designers

 Defines design workflow

Topology families
- Extended Generalized Fat Trees
- HyperX
User inputs:

- Number of servers, Number of racks and rack layout restrictions
- Bandwidth, Hop count
- Available parts (switches, cables, racks) and cost models
- One or more of topology families
Topology planning workflow

Candidate logical topology generation:
- Extended Generalized Fat Tree (EGFT) - Covered in this talk
- HyperX - See paper
- Our framework allows plugging in other topology generators
EGFT topology

Extended Generalized Fat Tree topologies

Parameters:

- Number of levels, $L$
- Aggregation factor at each level, $M_l$ for $1 \leq l \leq L$
- Number of top switches in each module at each level, $C_l$ for $1 \leq l \leq L$
- Number of links from top switch to each module, $K_l$ for $1 \leq l \leq L$

For $l = 3$:
- $M_3 = 2$
- $C_3 = 2$
- $K_3 = 2$
- $O_3 = 1$

For $l = 2$:
- $M_2 = 2$
- $C_2 = 2$
- $K_2 = 1$
- $O_2 = 1$

For $l = 1$:
- $M_1 = 2$
- $C_1 = 1$
- $K_1 = 1$
- $O_1 = 1$

Oversubscription
Generating Candidate Topologies: EGFT

Bottom-up exhaustive search

- Given: N servers and R-port switches
- For each level \( \ell \), choose \( M_\ell \), \( C_\ell \), \( K_\ell \)
- Requirement:
  Each top switch should connect to all \( M_\ell \) level \((\ell - 1)\) modules
- Constraints:
  \[
  M_\ell \leq R \\
  C_\ell \leq \text{number of free ports at level } (\ell - 1) \text{ module } = f_{\ell-1} \\
  K_\ell \leq R/M_\ell \quad \text{AND} \quad K_\ell \leq f_{\ell-1}/C_\ell
  \]

Search space can be huge

- Example: With \( N=1024 \) and \( R=48 \), size > 1 billion
EGFT: Heuristics to Prune Search Space

**H1:** At the top level, use the maximum lag factor possible

**H2:** Ignore all possibilities at a level that achieve lower oversubscription than at the lower levels

**H3:** If all lower level modules can be aggregated into one module, then do not consider other possible aggregations

**H4:** At the top level, use as many available switches as you can for the core switches
Effectiveness of EGFT Heuristics

The diagram shows the search space size for different configurations of #Terminals/Edge Switch, Switch Radix, and #Servers. The configurations include:

- 1K-48-16
- 1K-48-32
- 2K-48-16
- 2K-48-32

Legend:
- None
- H1
- H2
- H3
- H4
- All

The x-axis represents the configurations, and the y-axis shows the search space size (in log scale).

Note: The information contained herein is subject to change without notice.
Topology planning workflow

User inputs → Generate Candidate Topologies → Generate Physical Layout → Compute Design Cost → Visualize Results

<table>
<thead>
<tr>
<th>Logical Topology</th>
<th>Physical Layout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switches</td>
<td>Racks: rows, # racks/row</td>
</tr>
<tr>
<td>Servers</td>
<td>Positions for each Switch &amp; Server</td>
</tr>
<tr>
<td>Links</td>
<td>Type &amp; layout of cables</td>
</tr>
</tbody>
</table>

Heuristics:
- Avoid placing server and its edge-switch in two different racks
- Pack a rack tightly before using another rack
Part and manufacturing costs:

- **Switches**: $500 per 10GbE port
- **Cables and connectors**
  - Cost depends on the length and type of a cable
- **Cable installation labor**: $2.50 per intra-rack and $6.25 per inter-rack
- **Note**: Perseus can be used with other cost models
Topology planning workflow

Visualization: Rudimentary at this time
- Excel sheets
- 2-D plots
- DOT diagrams using GraphViz, an open source graph visualization package
Sample Results
Experimental parameters

Parameter values:

- Number of servers: 1024 to 8192
- Switch radices: 32, 48, 64, 96, and 144
  - Restrict to topologies with only single switch type
- Various number of terminals per switch

Disclaimer:

- Switch and cable costs are list prices; would be cheaper in bulk
Cost vs. Bisection BW: 1024 servers, FatTree

Switch Radix

Servers/Edge Switch

R = 96, T = 25
R = 48, T = 24
R = 144, T = 24
R = 96, T = 32
R = 64, T = 32
R = 64, T = 24
R = 96, T = 24
R = 48, T = 25
R = 144, T = 25
R = 64, T = 25
R = 48, T = 32
R = 144, T = 32

Over-sub

Bisection bandwidth (normalized)
Cost vs. Bisection BW: 1024 servers, FatTree

Ease of upgrade:
R=64 and T=32 is easily upgradeable to full-bisection bandwidth where as R=48 and T=32 only to 2:1 oversubscription
Cost vs. Bisection BW: 1024 servers, HyperX

For same number of switches, a different HyperX configuration can result in better bisection bandwidth at lower cost.
Further Steps
Optimization Problem: Logical to physical mapping

Problem: Given logical topology of switches, servers, and links, generate a feasible mapping of these onto a physical space with racks arranged in rows with multiple racks per row such that the wiring cost is minimized.

Rack constraints:
- Racks have fixed heights
- Limit on number of cables exiting a rack

Cable tray constraints:
- Each row has a cable tray running on top
- Not every column has a cross tray running on top, for cooling reasons

Cable constraints:
- Cheap copper cables have a maximum span (about 10 meters)
- Expensive optical components need to be used for longer links
Other interesting optimization challenges

Performance metrics and costs not addressed currently:
- Non-uniform Bisection Bandwidth
- Reliability
- Expandability
- Serviceability: Maintenance, SKUs
- Power
- Topologies with different switch types

Topologies:
- BCube, CamCube, etc.:
  - Servers with multi-interface NICs
  - Servers acting as end-points and switches
Perseus Tool

Current status: a preliminary prototype

Further work:

- Scalability to design networks for 100K servers
  - Current heuristics allow scaling to 8-32K servers
- Visualization
- Generate wiring instructions
- Verify installations
Summary

Data-center wiring – a rich research area with several hard and interesting problems
  • A complex problem for manual design

Our current work barely scratches this problem space
  • Perseus: A framework to help engineers in exploring the large design space
  • Considered various topologies: EGFT and HyperX
  • Exposed several interesting problems
  • Heuristics for reducing the huge design search space
Thank you